Integrated Energy Management System

Adrian Vlose

1 Introducere

Proiectul se concentreaza pe dezvoltarea unei aplicatii pentru monitorizarea si gestionarea energiei. Sistemul include vizualizarea utilizatorilor si dispozitivelor asociate fiecaruia, precum si colectarea si procesarea datelor de consum energetic. Frontend-ul este realizat in React, iar backend-ul este construit folosind Spring Boot, RabbitMQ si alte tehnologii. Datele sunt stocate in baze de date MySQL.

2 Structura aplicației

- Backend-ul este format dintr-un microserviciu care gestioneaza masuratorile primite de la dispozitive. Acesta proceseaza datele trimise prin RabbitMQ, calculeaza consumul orar de energie, pregateste datele pentru afisarea grafica si notifica utilizatorii in timp real prin WebSocket atunci cand consumul depaseste limita stabilita. Datele sunt salvate intr-o baza de date MySQL.
- Simulatorul este o aplicatie separata care citeste periodic valori dintr-un fisier CSV (‘sensor.csv‘) si transmite datele sub forma de mesaje JSON catre RabbitMQ. Fiecare mesaj include timestamp-ul local, device_id-ul unic si valoarea masurata.
- Toate componentele sunt salvate in Docker, prin intermediul Dockerfile-urilor si a unui fisier docker-compose. Rularea aplicatiei se face din consola, iar testarea se realizeaza dintr-o interfata React.

3 Implementare

1. Simulatorul a fost realizat intr-o aplicatie Spring Boot, unde se citeste fisierul CSV (‘sensor.csv‘). In interfata aplicatiei, utilizatorul poate apasa un buton pentru a porni simularea. Aceasta foloseste adnotarea ‘@Schedule‘ pentru a trimite mesaje in coada RabbitMQ la un interval de 1 secunda. Fiecare mesaj contine datele citite din CSV, precum timestamp-ul, device_id-ul si valoarea masurata.

2. Microserviciul de Monitoring este responsabil pentru citirea mesajelor din coada RabbitMQ (trimise de simulator). Acesta proceseaza fiecare masuratoare si, daca valoarea masurata impreuna cu media ultimelor 5 valori depaseste consumul orar permis, se trimite o notificare in interfata utilizatorului. In plus, acest microserviciu sincronizeaza modificarile din baza de date a celui de-al doilea microserviciu folosind RabbitMQ si un ‘topicExchange‘. Datele modificate sunt transmise printr-un ‘routing key‘ specific tipului de operatie efectuat (adaugare, actualizare sau stergere).Daca utilizatorul doreste in interfata sa vada anumite date despre simulari,el poate alege dintr-un calendar o zi si daca au fost simulari pentru acea zi el va putea observa consumurile pe baza unor grafice.

Figure 1: Grafice despre simulari

Figure 2: Notificarile in timp live

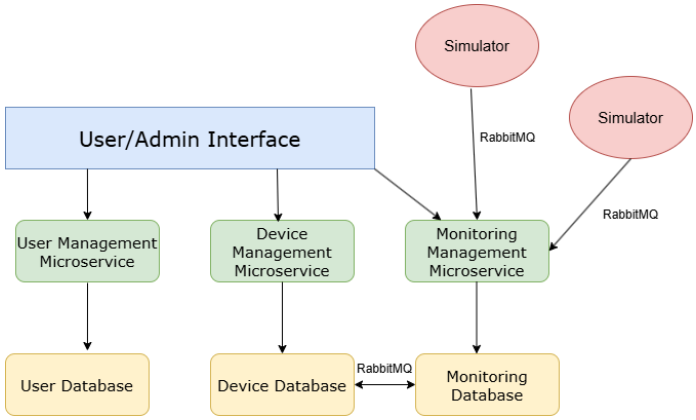

Figure 3: Diagrama arhitecturii conceptuale

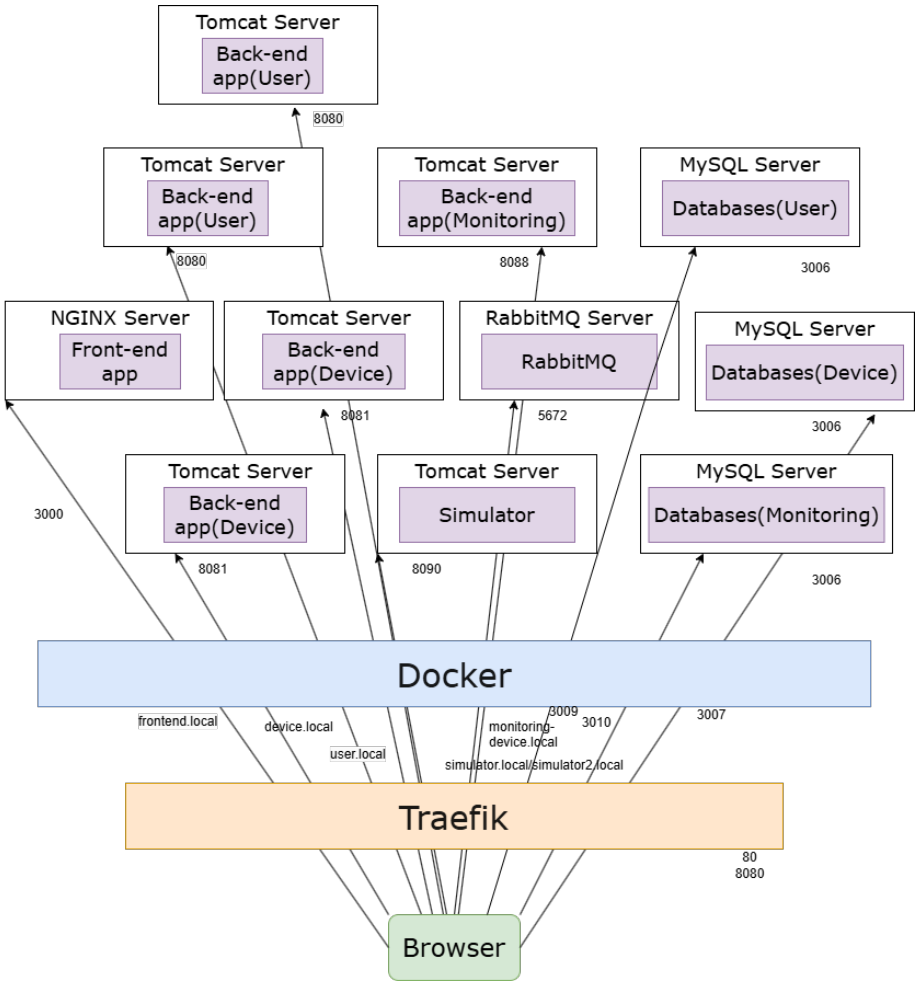

Figure 4: Diagrama de deployment

4 Concluzie

In concluzie, aplicatia dezvoltata pentru monitorizarea si gestionarea consumului energetic ofera o solutie integrata si eficienta. Utilizatorii beneficiaza de o interfata intuitiva, care le permite sa vizualizeze consumul energetic si sa primeasca notificari in timp real atunci cand depasesc limitele stabilite.

Microserviciul de Monitoring asigura procesarea rapida si precisa a datelor energetice primite de la simulator, sincronizarea automata a modificarilor din bazele de date si gestionarea eficienta a resurselor

prin RabbitMQ. Simulatorul permite testarea si simularea consumului energetic intr-un mod flexibil si realist, facilitand integrarea in infrastructura existenta.

Integrarea componentelor aplicatiei in Docker simplifica procesul de implementare si scalare, oferind flexibilitate si fiabilitate. Toate componentele au fost integrate prin Traefik si un network dedicat, asigurand o comunicare sigura si eficienta intre microservicii. De asemenea, utilizarea tehnologiilor moderne precum RabbitMQ si WebSocket imbunatateste performanta si interactivitatea sistemului, facandu-l o solutie adaptabila pentru nevoile complexe ale managementului energetic.